Quantifying the Value of IT-Investments

C. Verhoef
Free University of Amsterdam, Department of Mathematics and Computer Science,
De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands
x@cs.vu.nl

Abstract

We describe a method to quantify the value of investments in software systems. For that, we adopted the classical risk-adjusted discounted cash flow model and geared it towards the field of information technology. This resulted in a scenario-based approach incorporating two IT-specific risks that can substantially influence IT-appraisals. They are requirements creep and time compression. To account for the risk of failed IT-projects and overrun risks we proposed the Weighted Average Cost of Information Technology (WACIT). WACIT adjusts the well-known Weighted Average Cost of Capital, commonly used in discounted cash flow models. We proposed several methods to approximate WACIT ranging from an investment-specific rate to a (company-wide) one inferred from a quantitative IT-portfolio analysis. We illustrated our quantitative IT-investment management approach by way of a published example. For various risk-scenarios, we calculated using this example the impact on the firm's share price, standard economic indicators like NPV, IRR, PBP, ROI, RAROC, and more, in order to come to grips with the appraisal of information technology—the largest production factor of today.

Keywords and Phrases: valuation, discounted cash flow (DCF), net present value (NPV), internal rate of return (IRR), payback period (PBP), return on investment (ROI), risk-adjusted return on capital (RAROC), weighted average cost of information technology (WACIT), IT-investment management, quantitative IT-portfolio management, total cost of ownership (TCO), requirements creep, time compression, failure risk, overrun risk.

INTRODUCTION

Anyone with a background in economics knows that there are established tools and techniques to determine the value of proposed investments. Some investments are harder to appraise than others, but continuous research in financial economics is carried out to increase insight in such hard cases. Normally, a business case is valuated using a *discounted cash flow* (DCF) analysis. Roughly, the cost are subtracted from the benefits, time value for money is taken into account, and risks are dealt with. The outcome of such an analysis is the *net present value* (NPV), giving the net value in terms of today's money. If the NPV is positive, and a few other indicators are favorable, for instance the *pay back period* (PBP) is not too long, the *internal rate of return* (IRR) is not too close to the used discount rate, and if the appropriate variant of the *return on investment* (ROI) is not too low, you can decide to invest.

Of course, such economic indicators are a model only, and do not need to reflect reality accurately. For instance, during the dotcom hype a loss-generating free-software reseller could have a higher market value than the entire German automotive industry. For those hard cases additional analysis is required, for instance using *real option valuation* (ROV), where despite a zero NPV still a positive appraisal of an investment can be found due to the value that options can represent. For instance, the stock price of Netscape at the *initial public offering* (IPO), could be valued at $14 using a very optimistic DCF appraisal, but was set to $28 by its major underwriters. Via an ROV-augmented DCF it turned out to be possible to justify this higher value [4].

The problem While financial economics is well-established and applied in many areas, why do we hardly ever see quantitative financial analyses for major IT-investments? After all, not every IT-investment can be as complicated as the IPO of Netscape. The answer is that you need data for such analyses, which seems impossible to determine for IT-investment proposals. There are a few causes for that. For a start almost all organizations have a completely immature level of IT-development and maintenance. About 75% are still at the ad hoc level, and 15% are able to repeat a certain development process [21, p. 31]. At these levels, there is no overall metrics program; this prevents a meaningful data analysis. With such an analysis, you could for instance reveal the ratio of failures to successes, and with that determine the return that a success must have in order to compensate for the failed investments. The obvious solution is to establish a full-blown metrics program, but this fails in 80% of the cases [30]. On top of that, working with software is not a licensed profession, so that anyone who gets hired can join in managing, building, and maintaining information technology. In such a situation, it is not a surprise that there is no stream of (historical) data enabling quantitative insights, so that a rational IT-investment management practice could have emerged in the past.

The last few years IT-investments became so substantial that they are attracting the attention of executive management. For instance, it turned out among the large Dutch banks that 22% of their operational costs were attributable to information technology [8]. In fact, IT has become the main production factor for the financial industry, and this pattern is similar in other industries. Moreover, every few years you see the IT-budgets increase with double digit percentages, so it is time for proper IT-investment management. But there is no data, no hope for rapid sustainable improvement, so the worst executive nightmare imaginable, when you need to come to grips with the largest production factor of the firm: information technology. The author is advising large organizations on these problems, and has developed tools and techniques to alleviate some of the problems. The surrogates for real data—public benchmarks—are utilized to obtain an idea of cost, duration, risk, return, and financing of IT-investments, and on these surrogates we base the DCF-appraisals that are necessary to rationalize IT-decision making.

Goal In this paper we will show how to deploy financial techniques to quantify the value of IT-investments. We will focus on tailor-made software, and not on computer hardware investments, networks, or package acquisitions. We use an existing running example to illustrate the approach.

Related work It was hard to find a running example in the literature, since there is not much related work in this area. We mention the efforts by the General Accounting Office regarding ITIM, their IT Investment Maturity Model [27]. This work is complimentary to ours, in the sense that we dive into the technicalities of IT-investment management, and the GAO addresses how to embed this within an organization. Furthermore, we used an example from Gardner and Trotta [10, 9], but as we will see, this example makes assumptions on IT-development costs, operational costs, its lifespan, and more that are not in accordance with industry benchmarks. Then there is a textbook by Gardner on the value of information technology, which also does not take such issues into account [12]. It discusses networks, hardware, and other issues which are better understood than valuating custom-made IT-investments, which is our focus. Pisello and Strassmann [29] wrote an e-book, which creates awareness for the economic perspective in information technology. It gives an overview of standard economic performance metrics every CIO should know about. To a lesser extent this was also touched upon in the textbook by Boehm from the early 1980s, where net value, marginal analysis, time value for money, and discounted cash flows were briefly mentioned [3]. The focus of this book is on models for software cost estimation, not valuation. This paper dives much deeper into valuation questions, taking IT-specific aspects into account via benchmarked relations. Our work builds on a body of work by the author and his co-workers. We mention an elaborate publication on quantitative IT-portfolio management [35], recent work on quantitative aspects of outsourcing deals [36], and a few related contributions [34, 25, 24].

Organization of the paper In the next section we will discuss the running example, and derive via benchmarks data on the size of the software system. From the size estimate we calculate benchmarked key indicators: development costs, duration, risk, operational costs, and lifespan. Then we discuss discount rates, taking IT-specific risks into account. Subsequently, we devote a section to two other risks: requirements creep and time compression, and their combination, leading to 4 scenarios. In the next section we use the derived information to carry out risk-adjusted discounted cash flow analyses for the original example and the 4 scenarios. Next, we discuss the internal rate of return, the payback period, cost-benefit curves, break even points, return on investment, its risk-adjusted version RAROC, and more for the original example and the four scenarios. Finally, we vary the business case itself by discounting the cash flows that contain a benefit part. Then, we conclude and provide references to the literature.

RUNNING EXAMPLE

There are not many public documents giving insight into prospective IT-investments. Fortunately, *CIO Insight*—a business magazine—published a nice IT-investment example [10, 9]. We will use and adapt this example to illustrate how you can measure the impact on the firm's share price for a potential IT-investment. We quote the introduction of the example in its entirety:

> Amalgamated Widgets, a fictitious manufacturing company with $500 million in annual revenues, was under pressure from its shareholders and from Wall Street to increase its profitability. One possibility considered by top management: Install an online procurement system to cut purchasing costs for just about everything the company buys, from raw materials for its widgets to such commoditized items as safety helmets, work gloves and office supplies, on which the company spends a total of about $225 million a year. There's been a lot of hype surrounding e-purchasing, and the

> company's executives, ordinarily a pretty conservative bunch, were uneasy about making a significant investment in a new—and, in their minds, unproven—technology. So they asked their CIO and CFO a simple question: How much actual value might such a system bring to Amalgamated's shareholders?

Then the paper explains a standard discounted cash-flow model. The data that were used for this calculation were as follows. The IT-investment was outsourced at a price of $1.2 million dollar. The development time is not given, but in the case study we read that:

> Amalgamated's new system would not be up and running for a year because of the time it would take to build and install the system, train the company's purchasing employees, connect the system to suppliers and educate them in its use.

So, development time of the IT-system is apparently not more than 12 months. The operational lifespan of the ensuing system was estimated to be 4 years, and the operational costs were fixed at $610,000 per annum. Additional costs for migration or retirement are for the outsourcer. The cost savings of the project were estimated at zero in the first operational year, $850,000 in the second, $3.5 million in year 3, and $7.4 million in the fourth year [9]. Tax rates are 32%, and the investment cost is deducted over the operational lifetime of the project: a tax break of $96000 per annum (32% of $1.2 million, divided by 4 years). The discount rate that was used is 20%: 10% for the Weighted Average Cost of Capital (WACC), and another 10%, since the history has been that only one in two IT projects at Amalgamated resulted in a return [10]. The risk-adjusted discounted cash flow model resulted in a Net Present Value (NPV) of $2382686, which are the investment's net savings, discounted in today's money using the 20% discount rate. Another given is that there are 23.5 million shares outstanding which implies a positive impact of $0.10 per share, or since the share price was $10, a 1% increase on a single share (we will explain the calculations made later on).

Estimating the software size

A reliable key metric in economic studies on information technology is the function point metric [1, 7, 11, 23, 22]. There are many and diverse ways to recover this metric from information such as the minimal data of our running example. For now, it is not important to understand what a function point exactly comprises, other than that it is a synthetic metric giving an idea of the size of an IT-system. The function point metric can then be used as an IT-currency converter for our economic analyses. For our running example we will show how we can recover the amount of function points. Since we have no full information, we use public benchmarks for recovery of the lacking data. For the sake of this example we assume that the daily burdened rate of the outsourcer is $1000, and that there are 200 working days in a year, each day is eight hours of work. Then the amount of working hours for a $1.2 million IT-investment amounts to 9600. Using this number, we can estimate the amount of function points as follows. First we must find some public benchmarks on average productivity of outsourcers. In [21, p. 274] we find that for 100 function point systems the average work hours per function point is 4.33 hour. For 1000 function point systems this amounts to 10.41 working hours per function point, and for 10000 function point systems this is 27.39 work hours per function point. We used standard parametric statistical techniques to fit a smooth curve through these three benchmarks. Using an implementation in Splus [33, 26] of a nonlinear least squares regression algorithm [5, 17, 33, 28], the three observations can be fitted to the following curve:

$$hfp(f) \quad = \quad 0.6141603 \cdot f^{0.4121902} \tag{1}$$

In this equation *hfp* is short for *hours per function point*, and f is the amount of function points. So for a given amount of function points, hfp returns the average work hours per function point. This formula is not a perfect fit, but the residual sum of squares is 0.08664753 (zero represents a perfect fit). Indeed, $hfp(100) = 4.098851$ which is 0.23 off the original data point, $hfp(1000) = 10.588937$, which is 0.18 besides the second observation, and finally $hfp(10000) = 27.355369$, which differs only 0.03 from the original data point. Now the total effort is the amount of function points times formula 1: $f \cdot hfp(f) = 9600$. Solving this equation leads to an IT-investment of 932.9265 function points according to benchmark. Note that when you are to invest substantially in IT in an outsourcing context, formula 1 should not be used to settle your contract, but for this example it will do. For elaborate information on closing outsourcing deals we refer the interested reader to [36]. This does not dismiss us from the task of checking whether the found amount makes sense.

Sanity check As a first sanity check we deploy the following benchmark: $f^{0.38} = d$, where the power 0.38 is specific for outsourcing contexts [20, p. 202]. So, for a maximal development schedule of 12 months, we then find: 691 function points. A second sanity check is to use an empirical table [20, p. 185], where scope, class and type of the system are given a rating, and their sum to the power of 2.35 gives a rough indication of the amount of function points. For the scope we estimate it to be a stand-alone program (rate = 6), for the class, we take outsource contract (rate = 13), and the type is interactive with a GUI (rate = 5). Summing the rates yields 24, and to the power of 2.35 we find 1752 function points. Finally, we size by analogy: in an empirical table [20, p. 189], the closest we can find is a sales support system of type MIS, to be used in a business. This has a size of 975 function points. In the same table we find that an order entry system takes 1250 function points. These checks confirm the order of magnitude of our estimate using formula 1.

Deriving key indicators

In the original example [10], a number of assumptions were made on the prospective IT-investment, and based on them, a discounted cash flow calculation was carried out. We will use benchmarks to derive from our recovered function point size the key indicators necessary for such valuations, and compare our findings to the original example's data.

For a start, we use a few formulas to estimate duration, and development costs derived from industry benchmarks. We took the first 3 formulas from [35, p. 64, formulas (49–51)]. Formula 5 is new.

$$p(f) \quad = \quad 2.63431 + 21.36569 \cdot e^{-0.01805819 f^{0.5248877}} \tag{2}$$

$$d(f) \quad = \quad \frac{165}{p_o(f)} \tag{3}$$

$$
\begin{aligned}
(4) \qquad tcd_1(f) &= \frac{rw}{12} \cdot \frac{f}{p(f)} \\
(5) \qquad tcd_2(f) &= \frac{rf}{8} \cdot hfp(f)
\end{aligned}
$$

Formula 2 takes an amount of function points, and returns the number of function points delivered per staff month, so $p(932.9265) = 13.74478$. Formula 3 returns the benchmarked duration in calendar months for a given amount of function points, so $d(932.9265) = 12.00456$ months. Formulas 4 and 5 are based on productivity benchmarks: formulas 2 and 1 respectively. Both formulas return the *total cost of development* for a given amount of function points. So, $tcd_1(932.9265) = 1131250$, which is a little less than the quoted investment amount of $1.2 million. Of course, the result of formula 5 will be exactly $1.2 million, since we used its inverse to recover the amount of function points: $tcd_2(932.9265) = 1200000$. Given the complexity of formula 4, we often use formula 5 for recovery activities, and formula 4 for checking the result. The original example assumes a shorter development schedule than we estimated using benchmarked formulas. Namely, development, installing, training employees and suppliers, and connecting the system to the suppliers side is done in 12 months. Note that the original example does not include the costs for these activities, which can be substantial if the number of users is large. Leaving out training cost can mean the difference between a profitable and a loss-making investment. But for the sake of this example, we assume these costs to be zero.

We also want to derive operational costs, and the life expectancy of the system. We do that using the following formulas taken from [36] and [19, p. 419] respectively.

$$
\begin{aligned}
(6) \qquad mco(f) &= \frac{wr}{750} \cdot f^{1.25} \\
(7) \qquad y(f) &= f^{0.25}
\end{aligned}
$$

The *minimal cost of operation mco*, takes an amount of function points, and returns the minimal operational costs over the entire operational lifetime, so $mco(932.9265) = \$1374922$. And this amount is spent in $y(932.9265) = 5.526649$ years. Notice the large difference between the annual 610000 dollar and our annual 248780.4 dollar. The reason for the high number is that it is a fixed price, including exit and migration costs. For operational costs this is at the very high side, and via rational negotiations this amount could have been less (see [36] for a way to rationalize outsourcing deals). We will use this lower amount of operational costs in our valuation calculations.

RISKS OF FAILED OR RUNAWAY PROJECTS

The original example contains a risk premium above the cost of capital of 10%. We will show how you can quantify this risk premium more accurately. First, we need an indication of the *chance of failed projects* (*cf*), and the *chance of late projects*, denoted *cl*. The following formulas are based on outsourcing benchmarks, and taken from [35, pp. 46 and 48, formulas (30) and (34)]:

$$
\begin{aligned}
(8) \qquad cf(f) &= 0.3300779 \cdot \left(1 - \exp\left(-0.003296665 \cdot f^{0.6784296}\right)\right) \\
(9) \qquad cl(f) &= 0.4018422 \cdot \left(1 - \exp\left(-0.009922029 \cdot f^{0.5657454}\right)\right)
\end{aligned}
$$

Both formulas take a function point total as input and return a number between zero and one as an output. If you multiply that number by 100% you obtain a percentage indicating the chance of failed or late projects. So, the chance of failure is $cf(932.9265) = 9.540125\%$, which is indeed almost 10%. And the chance of serious cost/time overruns is somewhat higher: $cl(932.9265) = 15.19656\%$.

Discount rates In discounted cash flow calculations, a discount rate is used. For many investments, it is known what the company-wide Weighted Average Cost of Capital (WACC) comprises. The cost of capital is based on the cost of equity, debt, market value of the equities and debts, the percentages of financing in equity and debt, and the corporate tax rate. The CFO can inform you what the WACC is, since this is an indispensable number for valuating investments. But it is also important information to shareholders and lenders to the firm. Since information technology can make or break a company these days, it is important for investors to understand a premium on the WACC for major IT-investments. After all, the failure rates of software projects are high: Standish Group reported that about 30% of software projects fail, 50% are twice as expensive, take twice as much time, and deliver half the functionality, and only 20% of the software projects are on time, within budget, and with the desired functionality [18, 13, 14, 15]. In absolute figures, it was estimated in 1995 that this costed in the USA $ 81 billion on failed projects, and another $59 billion on serious cost overruns. It is our experience that using the WACC for discounting IT-investments leads to an over-valuation of the IT-investment. Indeed, in the original example, the premium on the WACC is 10%, since only half the IT-systems actually had a positive return. But is ten per cent enough? Too high, too low? Pisello and Strassmann [29, p. 41] propose to add 0% for no-risk investments, 10–15% for low-risk ones, 15–30% for medium risks, and for high risk investments 30% or higher. But how do you know the risk category? To address these issues, we developed the WACIT.

WACIT Normally, you have a portfolio of IT-investments. Some of them will be successful, some of them not. This needs to be accounted for when setting overall thresholds on potential returns. The threshold for each investment needs to be at least the cost of capital (WACC), plus the average risks. These risks can be averaged over the entire portfolio, and this premium could be called the Weighted Average Cost of Information Technology (WACIT). Within companies it is often known what the average cost of capital comprises. However, it is not known what the WACIT comprises. Note that if you wish to calculate a risk-adjusted ROI, like RAROC, for a single IT-investment, you actually have to carry out an IT-portfolio analysis to estimate the WACIT, so that you can use the proper discount rate for DCF-appraisals. At the time of writing this paper, we noticed that IT-investments are sometimes appraised via WACC-based NPV calculations, which is too positive given the risk-profile of IT-investments.

For instance, suppose we have ten projects like the running example, and only half of them deliver a return of at least the profitability threshold of the firm. For the sake of the example we set the return on the failing systems and the WACC both to zero. Still, the net profits need to be much higher to compensate for the failures. Suppose the firm demands an annual return of $2 per invested dollar. For the ten IT-systems, only half will have a return. If that is indeed $2 per dollar, the overall profitability is zero: only one dollar per invested dollar. So the entire ten-system portfolio is then underperforming. In order to have the $2 net return, the 5 systems need to make $4 per invested dollar. So the WACIT is then 50%. In the ideal situation, you need a rich IT-portfolio database to accurately measure the WACIT. In practice, IT-portfolio databases do not contain many data points, so we need to approximate the WACIT. We do this as follows:

$$\text{WACIT}(P) = \frac{1}{|P|} \cdot \sum_{p \in P} cf(f_p) + cl(f_p) \tag{10}$$

Let us explain this formula. We denote the IT-portfolio with P, $|P|$ the number of records in the portfolio, and p represents the information on an individual IT-asset, an going IT-project, or an IT-investment proposal. Normally this comprises of 3 data points: initiation date, delivery date, and cost (planned or actual, or both). Using recovery techniques, like the one we showed with formula 1 (or other techniques, see [35, 36]), we recover function point totals. We denote f_p to be the function point total we recovered from the information in record p. Then we calculate for each function point total in the portfolio the chance of failure and serious overruns. We can use formula 8, if it concerns an outsourced project, or similar formulas depending on the context. For example, we can use an MIS benchmarked failure risk formula as published in [35, p. 45, formula (28)], for business applications. If we cannot recover function points, we can use alternative formulas in terms of project duration (see, e.g. [35, p. 46, formulas (29) and (31)]). Similarly, we calculate for each element in the IT-portfolio the chance of late projects. Then we take the average, by summing and dividing by the number of entries in the IT-portfolio database. You can use this number as a premium on top of the WACC. So the discount rate to use in discounted cash flows is $\text{WACC} + \text{WACIT}(P)$.

If you cannot perform a quantitative IT-portfolio analysis to approximate the WACIT, here are some alternative techniques. If you are to act risk-averse, you could use the 80% as indicated by Standish, but we have never seen WACITs that high. You can also use different approximations. For instance, the average size of in-house built MIS systems is 950 function points [21, p. 185], and the chance of failure for such projects is 16.48714%, and the chance of serious cost overruns is 18.11008%. We calculated this using the formulas below (taken from [35]):

$$\begin{aligned} cf_i(f) &= 0.4805538 \cdot \left(1 - \exp\left(-0.007488905 \cdot f^{0.587375}\right)\right) && (11) \\ cl_i(f) &= 0.3672107 \cdot \left(1 - \exp\left(-0.01527202 \cdot f^{0.5535625}\right)\right) && (12) \end{aligned}$$

So, then the WACIT for an in-house IT-portfolio can be set to 34.59722%. We can do the same thing for an outsourced IT-portfolio. The average size is there 2750 function points [21, p. 269], which yields using formulas 8 and 9: $16.78399 + 23.44573 = 40.22972\%$. But of course, no IT-portfolio consists entirely of in-house, or outsourced projects, the portfolios we analyzed contain many small investment proposals and only a few very large ones. To give you an impression, in [35] a realistic IT-portfolio sample is given. This sample portfolio has a chance of failure of 13%, and a 14% chance of serious cost overruns [35, p. 49]. For that portfolio we take a WACIT of 27%. A final rule of thumb for approximating the WACIT is to calculate the individual risks of the investment, add them, and take that as the WACIT. We will do this in the running example, so the WACIT for that investment is: $9.540125 + 15.19656 = 24.73669\%$. For the WACC we assume the 10% as in the original example. When additional risks emerge, we adapt the WACIT with an extra premium. But, of course, it is better to carry out an IT-portfolio analysis, and calculate the overall WACIT.

REQUIREMENTS CREEP & TIME COMPRESSION

We have now discounted for failures and cost/time overruns. But there are more IT-specific risks that can influence the value of an IT-investment significantly. These risks are the requirements creep risk, and the time compression risk. You can choose to incorporate them into the WACIT, by doing an IT-portfolio analysis to estimate the requirements creep risk at the portfolio level, and the portfolio time compression exposure. But these risks can also accumulate, making the portfolio calculations hard. Instead, we opt for a scenario-based valuation, where we conduct a discounted cash flow analysis for 4 scenarios: one in the absence of requirements creep and time compression, one for each separate risk, and one when both risks are present. Moreover since the risk of failure and cost overruns increases when time compression and/or requirements creep play a role, we add a 10% to the WACIT.

KPIs	none	time comp.	req. creep	both risks
d	12.00456	11.00	12.43987	11.00
wh	9600	13289.13	11449.56	18095.74
tcd_1	1131250	1592391	1328041	2158814
tcd_2	1200000	1661141	1431195	2261968
mco	1374922	same	1606969	same
FP	932.9265	same	1056.893	same
y	5.526649	same	5.701745	same
cf	9.540125	same	10.23667	same
cl	15.19656	same	16.04986	same
WACIT	24.73669	+10%	26.28653	+10%

Table 1: Various scenarios for an IT-investment.

No additional risks The estimates that we have conducted so far are summarized in the second column of Table 1. The d, stands for the duration in calendar months, *wh*, is short for working hours, both *tcd*s are outcomes of formulas 4 and 5, *mco* is calculated with formula 6, FP is short for function points, y is the number of operational years calculated with formula 7, and the chance of failure (*cf*), is calculated with formula 8, and the chance of runaway projects (*cl*) is calculated with formula 9. Finally, we sum both risks, to approximate the WACIT.

Time compression risk Time compression of a software project is trying to do more work in a time frame than you would normally do from a pure technology viewpoint. We can quantify this risk using the following relation between time and effort, which is taken from [32, 31]:

$$e \cdot d^{3.721} = constant \tag{13}$$

where e stands for effort, and d is again the duration of a project. We use this empirically found relation to estimate the effect of a shorter deadline than would be normal according to benchmark. In Table 1 we can find the numbers for the normal effort and duration. From that we can calculate the constant, let's call it $c = 9600 \cdot 12.00456^{3.721} = 99659480$. Using this constant c we can estimate the effort if the duration of the development is different than our technology-driven estimate. Recall that the running example assumed that not only development, but also installing, training, connecting, etc were done in 12 months. Therefore, the time to implement the system is going to be less than 12 months. We conservatively set it to 11 months. Let us now calculate the increase in effort by having an 11 month deadline. This is calculated as follows: $e = 99659480/11^{3.721} = 13289.13$ working hours. So, 3689.13 additional working hours are required, just to make the deadline of 11 months. The cost per working hour is 125 dollar, which implies an additional cost of 461141.2, yielding a total cost of development of 1,592,391 for the first estimation method, and $1,661,141 if we use the development cost estimation formula 5. We know that the operational costs will probably be higher, since rushing an IT-project seldomly increases the quality of the delivered product. To compensate for higher maintenance costs, more delivered defects per function point, and other risks, we add 10% to the discount rate for a time compression-adjusted WACIT.

Requirements creep This is the addition of features after the requirements engineering phase is finalized. This is so endemic in software construction that there are benchmarks per industry for monthly growth rates. In our running example this benchmark is fairly low, since in a contracting situation, a more formal process of change is common. Still, the average monthly rate of requirements creep for the outsource industry is 1.10% [21, p. 269]. For our running example this implies the following. We know that the total effort of the requirements phase for systems in the 1000 function point range is around 7% [20, p. 179], and we set the impact on the schedule at 5% of the time. Then there is $12.00456 \cdot 0.95$ months time for growing at a rate of 1.10%. This amounts to the following requirements creep-adjusted function point total:

$$932.9265 \cdot 1.011^{11.40433} = 1056.893$$

So, an increase of 123.9665 function points. This increase has an impact on the development schedule, number of working hours, cost of development, operational cost, the lifespan, the risk of failure, risk on serious overruns, and can induce other risks. We summarized the impacts in Table 1 under the scenario name *req. creep*. The numbers are found by filling out the formulas for the requirements-creep adjusted function point total.

Both risks When there is requirements creep, and the deadlines are firm, usually there will also be an exposure to time compression. We again set the development schedule to 11 months. We calculated the numbers as follows: we used the requirements creep estimates and calculated via equation 13 the additional working hours and development cost. And we added ten percent to the requirements creep WACIT to adjust for the time compression.

Total cost of ownership (TCO) Since we have the development costs and the minimal cost of operation, we can estimate the minimal TCO. Although TCO is being seen as an important indicator to many, we think TCO itself is not the crucial issue. The crucial issues are in a TCO versus benefits analysis: given the total costs, and the total benefits, are we creating more benefits than incurring costs? And the second issue is also important: when are we earning/investing and how much? You could compare this to the TCO of a house: the cost of building it, and living in it can be calculated, this leads to the TCO of the house. But more important is how much mortgage you can afford given your income and potential future earnings. This monthly amount is what counts. In the end, you can add all the costs to calculate TCO. In IT-investment management, the time dimension is important to evaluate costs and benefits. This time value for money is expressed in risk-adjusted DCF models.

RISK-ADJUSTED DISCOUNTED CASH FLOWS

In Table 1 we calculated various alternatives, where we adjusted for the requirements creep risk, the time compression risk, and their combination. The cost difference between the least risky scenario and the most risky scenario is about 90%. This is why we separated requirements creep and time compression from failure and challenged projects. We need to know whether we should mitigate the risks, and this can be done via DCF appraisals: if the value is too low with a risk, but okay without it, it is worth mitigating it. Of course, this comes at a cost, too, but often a very low cost: a change control board, or a less tight deadline.

We need the business case to obtain an idea of the value of the 4 scenarios. In this section we will use the business case of the original example for calculating the NPVs for the various scenarios.

	Y1	Y2	Y3	Y4
benefits	0	850000	3500000	7400000
ops.	610000	610000	610000	610000
gross earn.	-610000	240000	2890000	6790000
inv.	1200000			
tax break	96000	96000	96000	96000
net taxes	-291200	-19200	828800	2076800
cash flows	-1518800	259200	2061200	4713200
rate	1.2	1.44	1.73	2.07
NPV	-1265667	180000	1191445	2276908

Table 2: Original discounted cash flows and NPV.

For the sake of comparison, we display the calculations of the original example [10, 9] in Table 2. In the first column the meaning of the numbers in the rows is explained. We have benefits, operational costs (ops.), gross earnings, tax breaks, net taxes, the cash flows, the discount rate of 20%, and the NPV. In the tables to come, we use the same notions in the first column. The other 4 columns show from year to year what happens to the numbers. In the first row we see the expected benefits. In year 1 (Y1), there are no benefits, since this is the investment period. Y2, gives a first small benefit, and this grows to almost ten times more in 2 years to $7.4 million. In the second row the operational costs are listed. The strange thing about this is that also during the software construction phase, where the system is not operational (hence the zero benefits), operational costs are taken into account. In our calculations we will leave out these costs during development, since they are not made. The next row shows the investment of $1.2 million, and no additional investments are necessary, so the rest of the columns are empty (meaning zero). The tax break is calculated as follows in this example: the tax rate is 32%, and for the investment this means a 384000 tax benefit over four years. This is amortized evenly over that time frame leading to an annual tax break of $96000. The net taxes are then calculated by taking 32% of the gross earnings and subtracting the tax break. For the first column this amounts to $-195200 - 96000 = -291200$. The other cells in this row are calculated analogously.

In the next row, we subtract the investment costs and the net tax from the gross earnings, yielding the cash flows: $-610000 - (-291200) - 1200000 = -1518800$, and so on. In the next row we give the discount factors: $(1 + 0.20)^n$, for $n = 1, \ldots, 4$ (in the original example these numbers are rounded to 2 digits). In the final row we adjust the cash flows with the rates calculated in the row above it: $-1518800/1.2 = -1265667$ etc. If we total the annual present value we obtain an overall NPV of $2382686. The example assumed 23.5 million outstanding shares which leads to a 10 cent impact per share.

time (yrs)	1.000	Y1	Y2	Y3	Y4	Y5	0.5266
benefits		850000	3500000	7400000	7400000	7400000	3897200
ops		-248780	-248780	-248780	-248780	-248780	-131020
gross earn.	0	601220	3251220	7151220	7151220	7151220	3766180
inv.	-1131250						
tax break	55462	55462	55462	55462	55462	55462	29209
net taxes	-55462	-247852	-1095852	-2343852	-2343852	-2343852	-1234387
cash flows	-1186712	353368	2155368	4807368	4807368	4807368	2531794
rate	1.348	1.816	2.446	3.296	4.441	5.984	7.001
NPV	-880664	194628	881078	1458528	1082502	803420	361635

Table 3: Discounted cash flows and NPV for scenario 1.

In Table 3, we display similar calculations as in the original example. Based on our quantitative analysis we found different numbers for development costs, different operational costs (much lower than the original example), a longer expected lifespan of the operational life of the proposed investment, and different discount rates. A longer lifespan implies potential longer benefits. For the sake of ease, we assumed a sustainable profit once the full cost savings of $7.4 million is reached in year 3.

The first row represents a time line in years, the second entry in this row is not a year, but is taken to be the development time (coincidentally a year), then we see five 12-month periods, and the final entry does not comprise a total year as well: 0.526649 of a year, which is the remaining fraction of the benchmarked lifespan for this scenario. We separated the investment period from the operational period. In the development period there are no operational costs, since the system is not operational. This means that these costs will only be taken into account after the system is operational. The tax benefits on the investment are amortized over the entire life cycle: development time plus operational phase. For the rates we used $(1+r)^{t+d/12}$, for $t = 0, \ldots, 5, y$; where d is the development time in months, r is the rate, and y is the operational lifetime.

time (yrs)	0.917	Y1	Y2	Y3	Y4	Y5	0.5266
benefits		850000	3500000	7400000	7400000	7400000	3897200
ops		-248780	-248780	-248780	-248780	-248780	-131020
gross earn.	0	601220	3251220	7151220	7151220	7151220	3766180
inv.	-1592391						
tax break	79084	79084	79084	79084	79084	79084	41650
net taxes	-79084	-271475	-1119475	-2367475	-2367475	-2367475	-1246827
cash flows	-1671475	329745	2131745	4783745	4783745	4783745	2519353
rate	1.403	2.031	2.940	4.255	6.159	8.914	10.831
NPV	-1190976	162331	725072	1124178	776706	536633	232611

Table 4: Discounted cash flows and NPV for scenario 2.

Tables 4–6, represent the same calculations as the scenario 1 case, but with differ-

time (yrs)	1.037	Y1	Y2	Y3	Y4	Y5	0.7017
benefits		850000	3500000	7400000	7400000	7400000	5192910
ops		-281838	-281838	-281838	-281838	-281838	-197778
gross earn.	0	568162	3218162	7118162	7118162	7118162	4995131
inv.	-1328041						
tax break	63067	63067	63067	63067	63067	63067	44257
net taxes	-63067	-244879	-1092879	-2340879	-2340879	-2340879	-1642699
cash flows	-1391109	323283	2125283	4777283	4777283	4777283	3352432
rate	1.378	1.879	2.560	3.489	4.755	6.481	8.054
NPV	-1009206	172087	830098	1369117	1004588	737114	416258

Table 5: Discounted cash flows and NPV for scenario 3.

time (yrs)	0.917	Y1	Y2	Y3	Y4	Y5	0.7017
benefits		850000	3500000	7400000	7400000	7400000	5192910
ops		-281838	-281838	-281838	-281838	-281838	-197778
gross earn.	0	568162	3218162	7118162	7118162	7118162	4995131
inv.	-2158814						
tax break	104379	104379	104379	104379	104379	104379	73247
net taxes	-104379	-286190	-1134190	-2382190	-2382190	-2382190	-1671689
cash flows	-2263192	281971	2083971	4735971	4735971	4735971	3323442
rate	1.417	2.073	3.033	4.437	6.490	9.494	12.399
NPV	-1596924	136008	687141	1067478	729717	498827	268039

Table 6: Discounted cash flows and NPV for scenario 4

ent input parameters. Table 7 summarizes the cumulative NPVs and their impact per outstanding share for the original example, and the 4 scenarios. The numbers in the second column are found by summation of the NPV rows in the previous tables. The third column is found by dividing the NPV by the number of outstanding shares (23.5 million).

Speed-to-market does not pay off Ranking the alternatives in NPV, we see that scenario 1 is the most valuable. There we mitigated requirements creep and time compression, by a technology driven deadline. Next in ranking is scenario 3: here we adjusted for requirements creep risks. The difference in NPV gives you an idea of how much money you can afford to invest in mitigating this risk. Normally, you can mitigate this risk at a low cost, by the installment of a formal chance control board, and formal change requests. If the size of the investment is larger than 1000 function points, you need additional investments to mitigate requirements creep like automated change control tools, Joint Application Design sessions for major modifications, requirements inspections, sizing major changes, planned evolution, formal cost estimates for major changes, and more [21, pp. 287–8].

The original example and scenario 2 (fourth rank) are very close, the original being slightly better off: ranked 3rd. We note that the lower operational costs and the longer operational phase (plus benefits), cancel out the higher fixed price contract and shorter life time. Obviously, if you not only negotiate a better price in your outsourcing deal (see [36] for help with that), but also take the life expectancy benchmarked operational phase into account, the investment scenario 1 is much more valuable than the original example.

The two lowest ranking scenarios, both suffer from time compression. The often heard business demand for speed-to-market does not pay off in this case, as our

	rank	cum. NPV	impact share price
original	3	2382686.633	0.101390894
scenario 1	1	3901127.931	0.166005444
scenario 2	4	2366555.157	0.100704475
scenario 3	2	3520056.631	0.149789644
scenario 4	5	1790285.471	0.07618236

Table 7: Net present value and share price impact for all scenarios.

quantitative analysis shows. Although the benefits come earlier, the incurred cost and additional risk turn the investment into the lowest ranking one if you look at NPV. This implies that when an IT-investment is time-critical, this will come at a cost. And maybe the cost and risks are so high, that a little bit more relaxed development schedule turns out to be much more valuable. Of course, when market share is at stake, the costs can be justified. The nice thing about analyses like the one we discuss in this paper is that you can now opt for a strategy: if lower value compensates for, say, loss of market share, then a time compression risk should be accounted for. Otherwise, we think it is better to slow down on development, and speed up on value.

INTERNAL RATE OF RETURN

	rate used	IRR
original	0.2	0.833688
scenario 1	0.3473669	1.32909
scenario 2	0.4473669	1.05109
scenario 3	0.3628653	1.18742
scenario 4	0.4628653	0.841652

Table 8: Used discount rate and internal rate of return for all scenarios.

The internal rate of return (IRR) is the discount rate such that the net present value in a discounted cash flow analysis is zero. You can use it to see how sensitive the used discount rate is for unforeseen risks. If the used rate is close to the IRR, this is riskier than if there is a certain gap. You can rate different investment proposals to their gap between IRR and the actual discount rate. Or you can formulate a simple IT-governance rule that the IRR should be 30% higher than the discount rate, but the exact numbers should be based on an IT-portfolio analysis of a mature IT-portfolio database.

For the original example, the IRR is 83.4%. We will show how to calculate this rate at which the invested amount of money is precisely expected to come back. For the original example this amounts to solving the following mathematical equation:

$$\frac{-1518800}{(1+r)} + \frac{259200}{(1+r)^2} + \frac{2061200}{(1+r)^3} + \frac{4713200}{(1+r)^4} = 0 \tag{14}$$

You can use a spread sheet program and find the rate r by trial and error. We used the computer algebra system Mathematica [37] to solve this equation. For that you have to give the computer algebra system a first guess. In all scenarios we used $r = 0.5$ as an initial value, which converged to a solution. The outcome of this exercise is $r = 0.833688$, so indeed 83.4%. We did not use pure years in the other four scenarios, but the development time d_i, for scenarios $1, \ldots, 4$, and in the last time period we used the expected operational lifetime: y_i, for scenarios $1, \ldots, 4$. As an example we give the equation for scenario 1:

$$\begin{aligned} \frac{-1186712}{(1+r)^{d_1/12}} + \frac{353368}{(1+r)^{d_1/12+1}} + \frac{2155368}{(1+r)^{d_1/12+2}} + \\ \frac{4807368}{(1+r)^{d_1/12+3}} + \frac{4807368}{(1+r)^{d_1/12+4}} + \frac{4807368}{(1+r)^{d_1/12+5}} + \\ \frac{2531794}{(1+r)^{d_1/12+y_1}} = 0 \end{aligned} \tag{15}$$

We solved equation 15 using Mathematica. The rate turns out to be 1.32909, so the IRR is 133%. In Table 8 we summarized the used discount rate, and the accompanying IRR for all the scenarios. The difference between the used discount rate and the IRR gives you an impression how much risk you can take before the net present value becomes negative. According to our simple IT-governance rule, all scenarios are okay. But of course, IRR is not the only indicator, and the next section shows another important one.

PAYBACK PERIOD

The payback period is the time it takes for an investment to become cash flow positive. It is an important economic indicator providing insight in the time it takes before value creation can commence at all. A related number is the break even point: the amount of money that is invested up to the time of the payback period. In [36], we defined the *payback period risk* which is the risk that the payback period is so long after the initial investment that the chance the environment has changed is substantial. This implies that you probably have to invest more before you can create value with the investment; this in turn will prolong the payback period even further, up to a point where a positive cash flow is never generated. Sometimes the phrase *hockey stick effect* is used for this phenomenon, referring to a J-curve on the side, so no positive cash flows will ever emerge (we will come back to J-curves later on). Mitigating this effect by only allowing short-term value-creating investments is usually not recommended, since long-term investments are opportunities enabling sustainable growth and business continuity. Therefore a balance between long-term and short-term investments is necessary. Finding such a balance is out of scope for this paper; it comprises an IT-portfolio analysis [35] and more.

We will calculate the payback periods of each scenario, and compare them in terms of the payback period risk. For the original example, there are no benefits when development ends, but one year after its launch, the benefits are $850000, and so on for the other years. At the end of year 4, the cumulative benefits are $11.8 million. Similarly, at the end of the development period (1 year in this case), we spent $1.2 million, plus 610000 on operational costs, accumulating to the end of year 4 into $3.64 million.

We display the cumulative amounts in Figure 1. Along the vertical axis we set out money in millions of dollars, and the horizontal axis represents elapsed time in

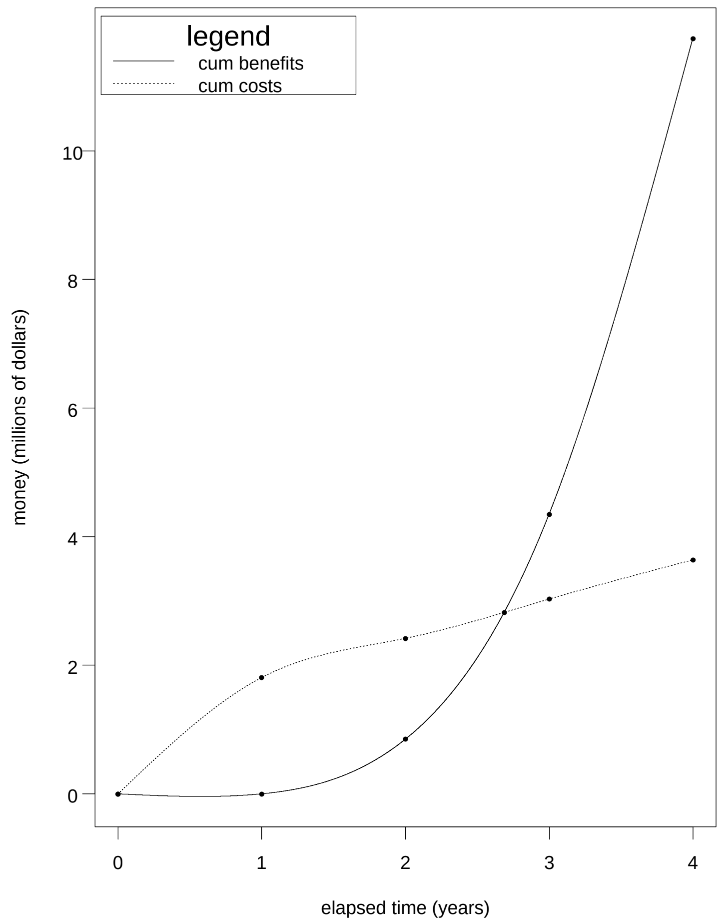

Figure 1: Visualizing the effort allocation over time for the original example.

years. For cash flow rates we assume a smooth continuous rate, which is, in our viewpoint, nicely approximated by cubic splines [2, 16]. As an alternative—easier—approximation you can connect the points in Figure 1 by line segments, e.g., using a simple spread sheet program (then you assume linear cash flow rates with intermediate jumps). No matter what approximation you choose, for comparison purposes you need to be consistent and apply one method for all scenarios. The solid spline in Figure 1 is the cumulative benefit curve, and the dashed one represents the cumulative costs. We used Splus [33, 26] to derive the cubic splines and to calculate the intersection point, which gives us the payback period and the break even point. For the original example the payback period is 2.7 years, and the break even point is $2.8 million.

An IT-portfolio analysis can reveal how large the risk of long payback periods is, and how it affects the overall created value at the portfolio level. From that you can derive guidelines with respect to payback periods and break even points. For instance, IT-governance rules regarding these aspects could look like this:

- The payback period should not be longer than half the entire risk-adjusted life cycle of an IT-investment. Risk-adjusted means that shorter or longer lifespans due to time compression and/or requirements creep should be taken into account.

- The break even point should not be more than twice the risk-free investment costs. Risk-free means the exclusion of the additional costs due to time compression and/or requirements creep.

According to the above IT-governance rules, the investment plan laid out in the original example should be sent back to the drawing board. For, the payback period is more than half the total lifespan, and the break even point is more than twice the initial investment ($1.2 million). So, having a positive NPV is not necessarily the sole decision making criterion.

Now let us look at these economic indicators for the other scenarios. In Figure 2 we derived cubic splines for scenario 1. The payback period for this scenario is 2.3 years, at a break even point of $1.4 million dollar. So we see that the lower and later operational costs and the longer expected lifespan contribute to a payback period and break even point that are in accordance with the above IT-governance rules.

	PBP	% of life cycle	break even (mn)	frac. risk-free inv.
original	2.68758	67.19	2.823	2.35
scenario 1	2.273006	34.82	1.431	1.26
scenario 2	2.361726	36.65	1.920	1.70
scenario 3	2.4067	35.72	1.694	1.50
scenario 4	2.550362	38.53	2.586	2.29

Table 9: Payback periods and the break even point for all scenarios.

We carried out these inferences for all scenarios, and summarized the results in Table 9. We also calculated the data necessary to decide whether or not the scenarios satisfy the IT-governance rules. The original example and scenario 4 (time compression and requirements creep) are not satisfying the criteria. These scenarios should not be approved. For instance, the original example needs a better outsourcing contract, and scenario 4 shows that you need to mitigate both exposures first.

To gain a little bit more insight, we depict in Figure 3 the connection between the break even points and payback periods of the different scenarios. So, for the *same* IT-investment you can either end up in the upper-right corner: the longest payback time and highest break even point. And with a different scenario, you can be at the lower-left corner: the fastest payback and lowest break even point. The original example ends up in the bad corner, scenario 1 in the good corner, and the other three scenarios somewhere in between. The numbers attached to the dots in the curve represent the different scenarios. Depending on the business landscape you can opt for a good, bad or ugly scenario. But then at least you have an idea of the potential economic implications.

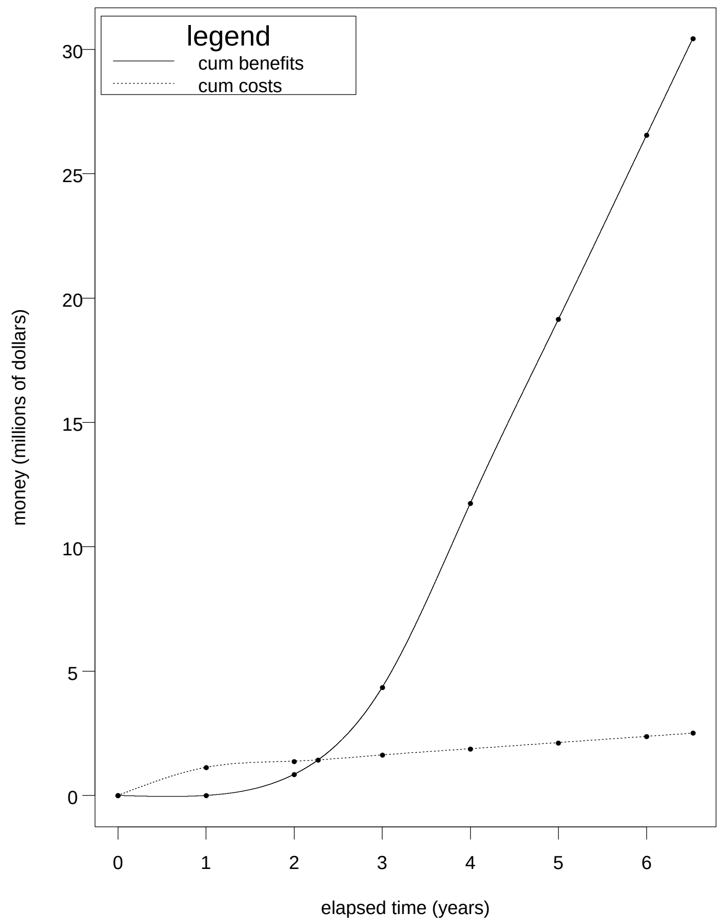

Figure 2: Visualizing the effort allocation over time for scenario 1.

J-curves

If we subtract the benefit spline from the cost spline, we obtain the characteristic J-curve: showing the net cash-flow over time. The payback period is then the intersection of the J-curve with the horizontal axis. In Figure 4 we displayed the J-curves for all scenarios.

It is hard to see differences between the various J-curves—they all dive under zero for some time, and then resurrect and move up. Note that the solidly plotted J-curve ends at year 4. This is the original example. The other scenarios have longer tails, since a longer benchmarked life cycle was assumed. We already gave more insight in the payback periods by showing the serpentine line in Figure 3. It is also interesting to understand the maximum negative cash flow, and when the largest negative flow is due. This amounts to finding the minimum of a J-curve. We approximated these points

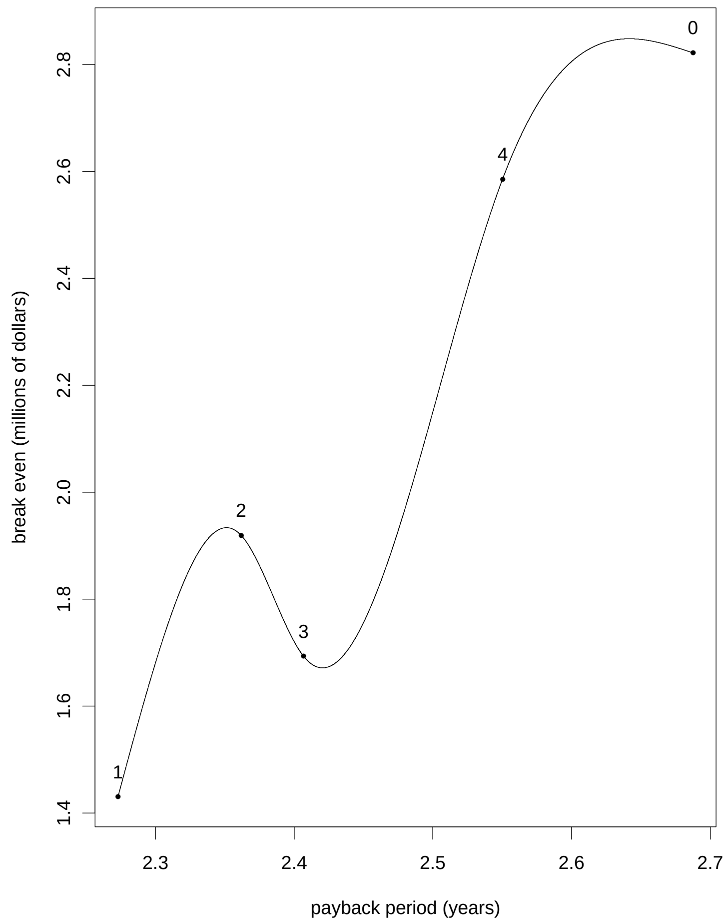

Figure 3: Break even distribution over payback period for all scenarios.

numerically using Splus. In Figure 4 we plot those points, and we summarized their numerical values in Table 10.

	minimum	time
original	-1990746	1.38587
scenario 1	-1176144	1.207866
scenario 2	-1684888	1.16649
scenario 3	-1386181	1.266477
scenario 4	-2304831	1.198008

Table 10: Maximum negative cash flows and when this is reached for all scenarios.

To gain more insight in these numbers, we plotted another serpentine line in Fig-

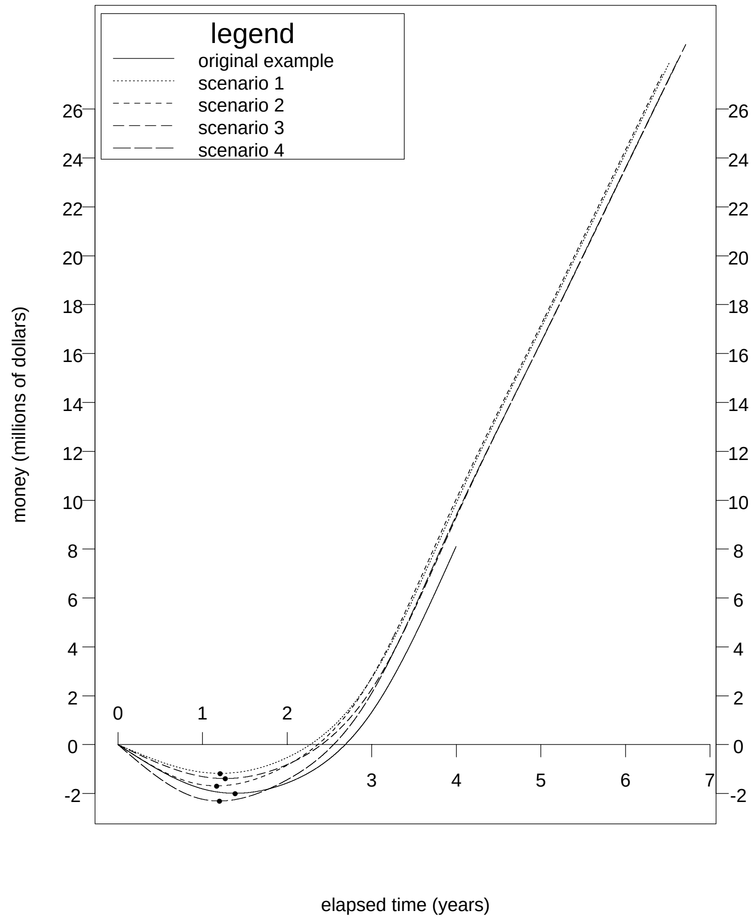

Figure 4: J-curves for all scenarios.

ure 5. This wavy line shows that for scenario 4 most money is burned in a short time frame: more than 2.3 million dollar negative, in 1.2 year. Indeed this scenario suffers from compounded requirements creep and time compression risks. Then the original example is next with a maximum negative flow of almost 2 million dollar. but it takes slightly more time to burn that. Then scenario 2 follows: about 1.7 million is the lowest point, which is like the other time compressed scenario burning the fastest of all other scenarios. Then follows scenario 3 with an even less negative minimum, taking more time than all the others, except the original example. Finally, scenario 1 has the lowest negative cash flow, which is reached at almost the same time as scenario 4. Knowing the low end of the negative cash flows is important, since when the business case turns out to be not as solid as expected, it can be the case that no or much less value creation will be created after the IT-investment has become operational. A very low negative cash flow takes then substantially more time to recover than otherwise, since you have

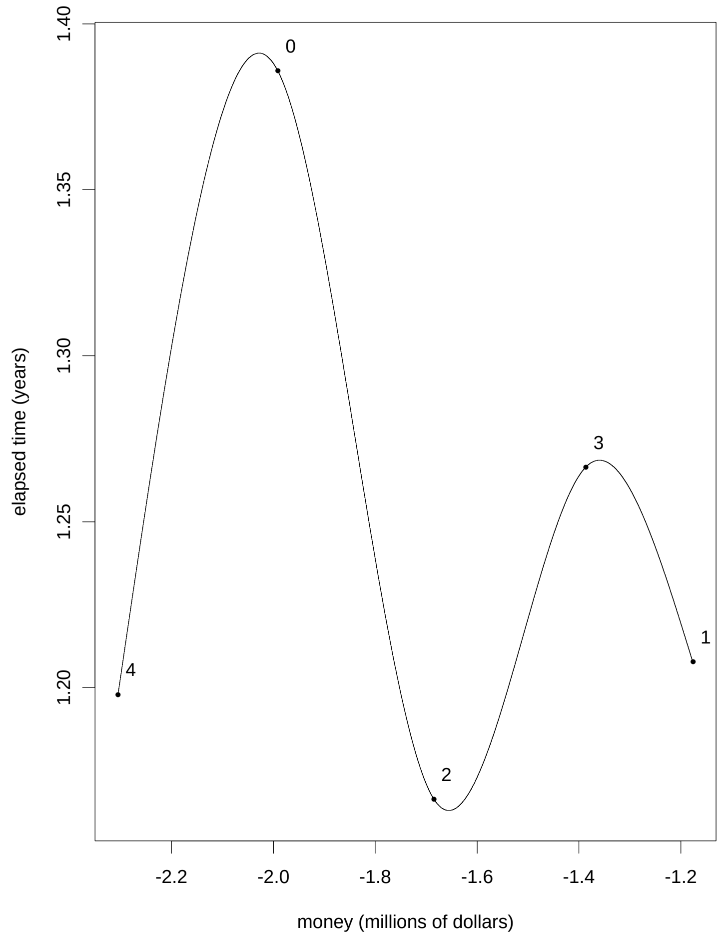

Figure 5: Distribution of minima of the J-curves for all scenarios.

to recover more costs.

RETURN ON INVESTMENT

The return on an investment (ROI) is defined as its net benefits divided by its total cost. Although this is a simple metric, there are more sophisticated variations of the classical definition necessary for IT-investments, due to the high risks involved.

You can risk-adjust benefits, the necessary capital, or both leading to the variants RAROC, RORAC, and RARORAC. RAROC means risk-adjusted return on (economic) capital. RORAC is the return on risk-adjusted (economic) capital. RARORAC is the combination: risk-adjusted return on risk-adjusted capital. We will restrict ourselves to a discussion of ROI and RAROC, and not dive into financial risks (is the money borrowed by a bank, or via a venture capitalist?), since this is out of scope for our paper. Needless to say that if an appraisal of an IT-investment proposal involves a financing strategy that bears financial risks deviating from the nominal WACC, you should risk-adjust for that, too.

	cum. net ben.	its NPV	cum. cost	its NPV	ROI	RAROC
original	8110000	2382687	3640000	2579232	2.23	0.92
scenario 1	27941028	3901128	2506172	1956014	11.15	1.99
scenario 2	27479887	2366555	2967313	2394245	9.26	0.99
scenario 3	28807899	3520057	2935010	2275028	9.82	1.55
scenario 4	27977127	1790285	3765783	3062465	7.43	0.58

Table 11: ROI and RAROC for all scenarios.

To calculate the ROI for the original example, we took the necessary data from Table 2. We accumulated the benefits (11750000), and costs (3640000), and subtracted them for the cumulative net benefit, which is $8110000. The NPV is also taken from this table: $2382687. We discounted the cumulative costs with a rate of 10%, leading to an NPV of the costs of $2579232. Now the ROI is the division of the cumulative net benefits and the cumulative costs: $8110000/3640000 = 2.23$. The risk-adjusted version that we use takes the NPV of the benefits, and the NPV of the costs, and divides that. This leads to a RAROC of $2382687/2579232 = 0.92$. In the original example a discount rate of 10% was used for the benefits, and we used the same rate for the costs. We summarized these numbers in Table 11.

For scenarios 1 to 4, we used a slightly different approach. We used different discount rates: for the NPV of the benefits we used the WACC plus the WACIT, and for the costs, we only used the WACC. This is because the benefits bear an IT-risk, whereas the cost of capital is assumed to satisfy the normal criteria of riskiness, so that we can use the WACC-rate. Of course, when the cost of capital is also bearing additional risk (maybe via venture capital, or expensive up-front financing constructs) the discount rate for capital should be adapted accordingly.

We summarized the results of these calculations for all the scenarios in Table 11. An issue that is worth mentioning, is that the difference between ROI and RAROC is substantial. This shows in a quantitative manner the risk profile of IT-investments. For scenario 1 this differs by a factor of more than 5. We also see that the original example has a very low RAROC, but still higher than the time compressed and creep exposed scenario 4. To gain more insight in the various scenarios we will show a connection between RAROC and the payback period.

RAROC – PBP We now connect the risk-adjusted returns with their respective payback periods. This leads to another viper trace as displayed in Figure 6. Obviously, you want the combination of the shortest payback period in combination with the highest return on investment. A short payback period bears a lower investment risk than a longer payback period, and if that can be combined with a high return, this is ideal. The scenario that fits this description best is scenario 1: with a RAROC of about 2, and a payback of about 2.3 year, this is the most favorable scenario. As we know, scenario 2 is a time compressed version of scenario 1. Scenario 2, halves the RAROC, and has a longer payback period. This unequivocally illustrates that this business case is not going to be more profitable by optimizing it to a speed-to-market scenario: the extra

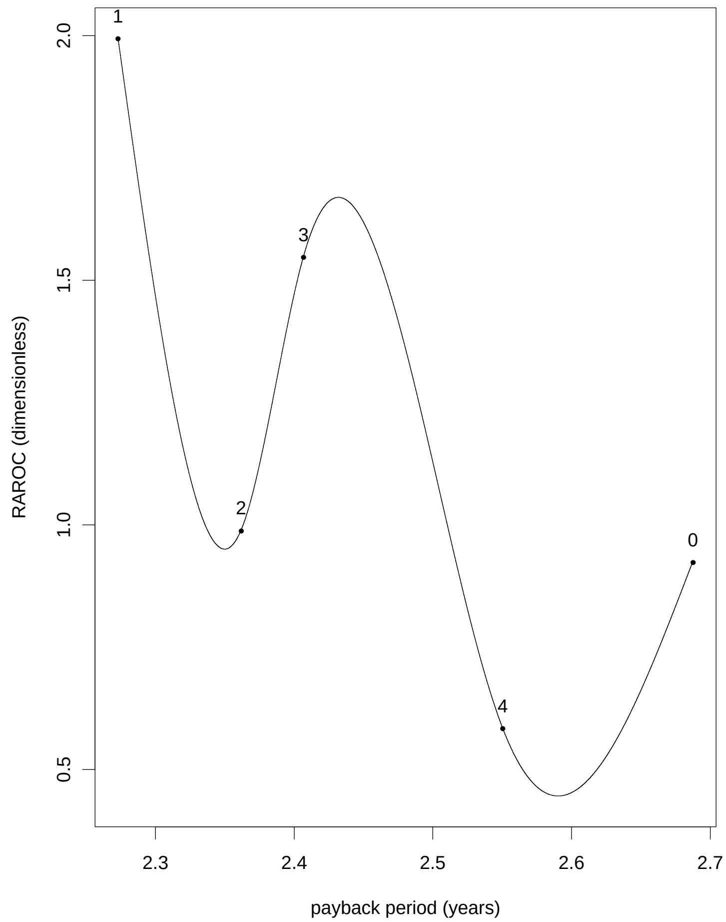

Figure 6: RAROC over payback period for all scenarios.

effort is not worth the lower benefits, which take longer to be generated. Of course, if there is no way out, you have to pursue this scenario, but then you at least understand the implications. Scenario 3 lowers the RAROC by about 25% compared to scenario 1, and also prolongs the payback period. So you see what the result is of not mitigating the requirements creep risk. Scenario 4 is the worst in the sense of the lowest RAROC: a little over 25% of scenario 1, but it does not have the longest payback. This is for the original example, with a RAROC a little less than half of the RAROC of scenario 1.

VARYING BUSINESS CASES

Our valuation approach is based on the cash-flow predictions by the business, and benchmarked relations regarding costs, durations, and risks. Until now, we investigated different risk scenarios and their impact on the value, by varying the IT-side of the equation. Indeed we found these scenarios will affect a valuation significantly. In this section we explain how we vary the business-side of an IT-valuation.

For a start, it is important to know the difference between the estimated benefits and the actuals so that we can adapt the proposed business case with this factor. We call this the *ROI IT-portfolio's organization-wide fantasy factor*, abbreviated RIPOFF. This is the factor with which the estimated benefits differ from the actual benefits. If you have historical data, and a good reporting system, you can measure the RIPOFF. Namely, the IT-investment plans project certain benefits, and by measuring the IT-performance of the materialized IT-assets, the actual benefits are known, and their quotient is the RIPOFF. Unfortunately, this is an ideal situation so you have to approximate the RIPOFF. For instance, by looking at a few large investments and their estimated and actual value creation. Or you can define a simple IT-governance rule stating that a business case should be beneficial even if we divide the benefits by 2 (but this has its own issues, as we will see later on).

Even without proper data, you want to know how sensitive an IT-investment is to lower returns than projected. To obtain an idea about this, we use a similar idea as calculating the internal rate of return. There we calculated the discount rate for which the NPV becomes zero, and compared it to the original discount rate. This gave an idea of the sensitivity to higher risks. So, you challenge the riskiness as perceived/used for constructing the information technology. Now we will challenge the business case: by discounting the cash flows with an amount of money such that the NPV becomes zero. So that we can see how sensitive the business case is for disappointing returns. To do this, we need to solve the following equation:

$$\sum_{i=1}^{n} \frac{c_i - R_i \cdot (t_i - t_{i-1})}{(1+r)^{t_i}} = 0 \tag{16}$$

In equation 16, there are $n+1$ time units $t_0, t_1, \ldots, t_n$, cash flows c_i, $i = 1, \ldots, n$, and r, is the used discount rate. For all i, where c_i contains nonzero benefits, we take $R_i = R$, and for the other i, we set $R_i = 0$. So, for the original example, equation 16 is instantiated as follows:

$$\frac{-1518800 - 0}{(1+r)} + \frac{259200 - R}{(1+r)^2} + \frac{2061200 - R}{(1+r)^3} + \frac{4713200 - R}{(1+r)^4} \quad = \quad 0 \tag{17}$$

The discount rate is $r = 0.2$. Now we solve equation 17 using Mathematica [37]. Therefore, we have to make a first guess, and we used a million dollar for R. The solution with this initial value for the root finding algorithm is then: $R = 1355879.6$. We correct for the tax rate leading to $1355879.6/(1-0.32) = 1993940.5$ million dollar per year. If we miss this amount of benefit per year the NPV will be zero. The estimated cumulative benefits for this investment are 11750000, and the zero NPV benefits are this amount minus 3 times the 1993940.5, which is 5768178.4 dollar. This means that the RIPOFF should be smaller than $11750000/5768178.4 = 2.037$. In other words, if the business is exaggerating with a RIPOFF of 2, and you correct for that in the business case, then the investment is doubtful.

For the first scenario, equation 16 looks similar:

$$\frac{-1186712 - 0}{(1+r)^{d_1/12}} + \frac{353368 - R}{(1+r)^{d_1/12+1}} + \frac{2155368 - R}{(1+r)^{d_1/12+2}} + \tag{18}$$

$$\frac{4807368-R}{(1+r)^{d_1/12+3}}+\frac{4807368-R}{(1+r)^{d_1/12+4}}+\frac{4807368-R}{(1+r)^{d_1/12+5}}+ \\ \frac{2531794-R\cdot 0.526648671}{(1+r)^{d_1/12+y_1}}=0$$

In equation 18 we use the discount rate $r = 0.3473669$. Again, using Mathematica, we find that $R = 2254359.1$ dollar, the tax corrected R is then 3315233.9 dollar. If we subtract this amount per year plus its appropriate fraction in the last year, the NPV will turn out to be exactly zero. Since there is 5.526648671 years of operational life, we have a cumulative R of 18322133.0 dollar. If we subtract that from the estimated cumulative benefits (30447200), we find 12125067 dollar, and the RIPOFF should not exceed $30447200/12125067 = 2.511$. So in this scenario, if the business is boosting its proposals with a fantasy factor of 2, then still this scenario will have a positive NPV, so this scenario is less sensitive to disappointing benefits, than the original example.

	cum ben.	R	tax corr. R	cum. R	max. RIPOFF
original	11750000	1355879.6	1993940.5	5981821.6	2.037
scenario 1	30447200	2254359.1	3315233.9	18322133.0	2.511
scenario 2	30447200	1701834.1	2502697.3	13831528.5	1.832
scenario 3	31742910	2118989.9	3116161.7	17767557.9	2.271
scenario 4	31742910	1322733.8	1945196.8	11091015.2	1.537

Table 12: The maximal RIPOFF for all scenarios.

We summarized the results of the maximally permissible RIPOFFs for all scenarios in Table 12. According our fictitious IT-governance rule that the zero NPV RIPOFF should not exceed 2 you can conclude that scenarios 1 and 3 are acceptable, that the original example is at the edge of being acceptable, and that scenarios 2 and 4 are not acceptable.

Ripped off by RIPOFF An unwanted side-effect of using an IT-governance rule like a RIPOFF of two to discount for the business case, is that the business will correct for it. Namely, if you use a RIPOFF of 2 (divide the cumulative estimated benefits by two, and amortize that over the benefit-containing cash flows), the business will sooner or later adapt their estimates to compensate for this effect. Tom DeMarco [6] proposed the so-called *estimating quality factor*, or EQF, meant to reward people on the quality of their estimates rather than the estimates themselves. The idea is to divide the actual value (that we find in retrospect), by the difference with the estimate, and this over time. So, suppose that a turns out to be the actual value, and $e(t)$ represents the various estimates over time, and t_a is the time when the actual value is known, typically at the end of a project. Then the following calculation gives us the EQF:

$$\mathrm{EQF}(a) = \int_0^{t_a} \frac{1}{1-e(t)/a}dt \tag{19}$$

Equation 19 expresses that if the estimates are at all times exactly a, the estimating quality factor becomes infinite. So, in practice, the closer an estimate approximates the actual value, the higher the EQF will become. You could use this trick to prevent people from compensating for this governance rule, since it will in the end be exposed by the estimating quality factor for the RIPOFF, which is what the incentive is based on. Apart from this potential side-effect, we think that the RIPOFF calculations give a good impression of the sensitivity for a business case for disappointing returns.

CONCLUSIONS

In this paper we illustrated how you can appraise IT-investments. We explained our approach using an existing published IT-investment proposal, and we adapted and extended this example to illustrate how to obtain the most prominent quantitative aspects of estimating net tangible benefits of IT-investment plans. Our approach extends existing methods known from financial theory on the one hand, and on the other hand makes use of quantitative IT-portfolio management to supply important input to conduct the economic calculations. Importantly, based on minimal data points, we are still able to carry out a credible financial analysis of the costs, duration, and financing of IT-investments, based on benchmarks. Moreover, we use benchmarks to risk-adjust costs, durations, and financing of such investments. Thereby, we are in a position to provide insight into the standard economic indicators: net present value, internal rate of return, return on investment, its risk-adjusted variants, the payback period, break even points, cost-benefit analysis over time, J-curves, and comparisons of different risk scenarios. We also addressed the idea of discounting the projected benefits, and obtain an impression of how much lower the benefits can become before we have a bad investment. We think that this paper brings us one step closer to a common practice of IT-investment management.

References

[1] A.J. Albrecht. Measuring application development productivity. In *Proceedings of the Joint SHARE/GUIDE/IBM Application Development Symposium*, pages 83–92, 1979.

[2] E.K. Blum. *Numerical Analysis and Computation Theory and Practice.* Addison-Wesley, 1972.

[3] B. Boehm. *Software Engineering Economics.* Prentice Hall, 1981.

[4] A. Buckley, K. Tse, H. Rijken, and H. Eijgenhuijsen. Stock Market Valuation with Real Options: Lessons from Netscape. *European Management Journal*, 20(5):512–526, 2002.

[5] J.M. Chambers and T.J. Hastie, editors. *Statistical Models in S.* Wadsworth & Brooks/Cole, Pacific Grove, CA, 1992.

[6] T. DeMarco. *Controlling Software Projects – Management Measurement & Estimation*. Yourdon Press Computing Series, 1982.

[7] J.B. Dreger. *Function Point Analysis*. Prentice Hall, 1989.

[8] H.M.P. Kersten et al. Nederland in de grote dynamiek van electronisch betalingsverkeer: Demarreren of bijblijven? – Internationale ICT-toets bancaire sector 2000: electronisch betalingsverkeer. Technical report, CMG Finance, 2001. For the Ministry of Economic Affairs of The Netherlands. In Dutch.

[9] C. Gardner and R. Trotta. How Amalgamated Widget Determined Its Benefits Stream, 2002.

[10] C. Gardner and R. Trotta. How to Determine the Value of a Project, 2002.

[11] D. Garmus and D. Herron. *Function Point Analysis – Measurement Practices for Successful Software Projects*. Addison-Wesley, 2001.

[12] P.R. Garvey. *Probability Methods for Cost Uncertainty Analysis – A Systems Engineering Perspective.* Marcel Dekker Inc., 2000.

[13] The Standish Group. CHAOS, 1995. Retrievable via: standishgroup.com/visitor/chaos.htm (Current February 2001).

[14] The Standish Group. CHAOS: A Recipe for Success, 1999. Retrievable via: `www.pm2go.com/sample_research/chaos1998.pdf`.

[15] The Standish Group. EXTREME CHAOS, 2001. Purchase via: `https://secure.standishgroup.com/reports/reports.php`.

[16] R.W. Hamming. *Numerical Methods for Scientists and Engineers*. McGraw-Hill, 2nd edition, 1973.

[17] S. Huet and M.-A. Gruet. *Stastical Tools for Nonlinear Regression – A Practical Guide with S-Plus Examples*. Springer Verlag, 1996.

[18] J. Johnson. Chaos: The dollar drain of IT project failures. *Application Development Trends*, 2(1):41–47, 1995.

[19] C. Jones. *Assessment and Control of Software Risks*. Prentice-Hall, 1994.

[20] C. Jones. *Estimating Software Costs*. McGraw-Hill, 1998.

[21] C. Jones. *Software Assessments, Benchmarks, and Best Practices*. Information Technology Series. Addison-Wesley, 2000.

[22] C.F. Kemerer. Reliability of Function Points Measurement – A Field Experiment. *Communications of the ACM*, 36(2):85–97, 1993.

[23] C.F. Kemerer and B.S. Porter. Improving the Reliability of Function Point Measurement: An Empirical Study. *IEEE Transactions on Software Engineering*, SE-18(11):1011–1024, 1992.

[24] B. Kersten and C. Verhoef. IT Portfolio Management: A Banker's Perspective on IT. *Cutter IT Journal*, 16(4):34–40, 2003.

[25] S. Klusener and C. Verhoef. 9210: the zip code of another IT-soap, 2003. Available via `www.cs.vu.nl/~x/9210/9210.pdf`.

[26] A. Krause and M. Olson. *Basics of S and S-Plus*. Springer Verlag, 2nd edition, 2000.

[27] United States General Accounting Office. Information Technology Investment Management – A Framework for Assessing and Improving Process Maturity, 2000. Available via: `www.gao.gov/special.pubs/10_1_23.pdf`.

[28] J.C. Pinheiro and D.M. Bates. *Mixed-Effects Models in S and S-PLUS*. Springer Verlag, 2000.

[29] T. Pisello and P.A. Strassmann. *IT Value Chain Management – Maximizing the ROI from IT Investments*, volume D10 of *Digital Publications from The Information Economics Press*. The Information Economics Press, New Canaan, Connecticut, USA, 2003.

[30] D.R. Pitts. Metrics: Problem Solved? *Crosstalk*, 1997. Available via: `www.stsc.hill.af.mil/crosstalk/1997/dec/metrics.asp`.

[31] L.H. Putnam and W. Myers. *Measures for Excellence – Reliable Software on Time, Within Budget*. Yourdon Press Computing Series, 1992.

[32] L.H. Putnam and D.T. Putnam. A Data verification of the Software Fourth Power Trade-Off Law. In *Proceedings of the International Society of Parametric Analysts – Sixth Annual Conference*, volume III(I), pages 443–471, 1984.

[33] W.N. Venables and B.D. Ripley. *Modern Applied Statistics with S-PLUS*. Springer Verlag, 3rd edition, 1999.

[34] C. Verhoef. Getting on top of IT, 2002. Available via `www.cs.vu.nl/~x/top/top.pdf`.

[35] C. Verhoef. Quantitative IT Portfolio Management. *Science of Computer Programming*, 45(1):1–96, 2002.

[36] C. Verhoef. Quantitative Aspects of Outsourcing Deals. *Science of Computer Programming*, 2004. To Appear. Available via: `www.cs.vu.nl/~x/out/out.pdf`.

[37] S. Wolfram. *Mathematica Book*. Cambridge University Press, 4th edition, 1999.